Centre de recherche sur les transports (C.R.T.)
Centre for Research on Transportation
CRT-2006-09

SOLVING A PRODUCTION SCHEDULING PROBLEM AS A TIME-DEPENDENT TRAVELING SALESMAN PROBLEM

by

Gabriella Stecco
Jean-François Cordeau
Elena Moretti

April 2006

Université de Montréal
C.P. 6128, succursale Centre-ville, Montréal QC H3C 3J7 Canada
Téléphone: (514) 343-7575 / Télécopieur: (514) 343-7121

crt@crt.umontreal.ca
www.crt.umontreal.ca

HEC ÉCOLE POLYTECHNIQUE MONTRÉAL Université de Montréal

106-07230/

PUBLICATION
CRT-2006-09

SOLVING A PRODUCTION SCHEDULING PROBLEM AS A TIME-DEPENDENT TRAVELING SALESMAN PROBLEM

by

Gabriella Stecco[1]
Jean-François Cordeau[2]
Elena Moretti[1]

This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

1. Department of Applied Mathematics, University Ca' Foscari of Venice, Dorsoduro n. 3825/E, 30123 Venice, Italy (gabriellastecco,emoretti@unive.it)

2. Canada Research Chair in Distribution Management, HEC Montréal, 3000 Côte-Sainte-Catherine, Montréal, Canada H3T 2A7, and Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 (cordeau@crt.umontreal.ca)

Dépôt légal – Bibliothèque nationale du Québec
Bibliothèque nationale du Canada, 2006

Abstract

This paper introduces and compares several formulations of a production scheduling problem with sequence-dependent and time-dependent setup times on a single machine. The setup time is divided into two parts: one that can be performed at any time and another one that is restricted to be made outside of a given time interval. As a result, the setup time between two jobs is a function of the completion time of the first job. The problem can be formulated as a time-dependent traveling salesman problem, where the travel time between two nodes is a function of the departure time from the first node. We show that the resulting formulation can be strengthened to provide stronger linear programming relaxation lower bounds. We also introduce several families of valid inequalities which are used within a branch-and-cut algorithm. Computational experiments show that this algorithm can solve instances with up to 20 jobs within reasonable computing times.

Keywords: production scheduling, sequence-dependent, time-dependent, setup times, traveling salesman problem.

Résumé

Cet article présente et compare plusieurs formulations d'un problème d'ordonnancement dans lequel les temps de mise en course dépendent à la fois de la séquence de production et de l'heure de mise en course des tâches. Le temps de mise en course peut être séparé en deux parties: une partie qui peut être réalisée en tout temps et une autre qui est restreinte à être effectuée à l'extérieur d'un intervalle de temps donné. Le temps de mise en course entre deux tâches est donc une fonction de l'heure d'achèvement de la première de ces tâches. Le problème peut être formulé comme un problème du voyageur de commerce avec dépendance temporelle. Nous montrons que cette formulation peut être renforcée pour fournir de meilleures bornes inférieures. Nous introduisons aussi des inégalités valides qui sont utilisées à l'intérieur d'un algorithme de séparation et coupes. Des expériences de calcul montrent que l'algorithme peut résoudre des instances comportant jusqu'à 20 tâches en des temps de calcul raisonnables.

Mots-clés : ordonnancement de la production, séquencement, horaire, mise en course, problème du voyageur de commerce.

1 Introduction

This paper studies a production scheduling problem with sequence-dependent and time-dependent setup times on a single machine. This *Sequence- and Time-Dependent Scheduling Problem* (STDSP) arises, for instance, when some setups cannot be performed during a certain part of the day. We consider the problem where the transition between two jobs involves two setups: one that can be performed at any time (an *unrestricted setup*) and another one that is required to take place outside of a given time interval (a *restricted setup*). It is possible to divide the restricted setup into two parts: one performed before the forbidden interval and the other performed after this interval. In addition, the unrestricted setup can be performed before or after the restricted setup. To schedule a job it is thus necessary to know the completion time of its predecessor, which determines whether both setups can be performed consecutively without interruption or if there is an idle time due to the restricted setup. Hence, the setup time between two jobs is clearly a function of the time at which the first of these jobs is completed.

We consider a planning horizon of several periods (e.g., days) with a forbidden interval that repeats in the same position in each period. This problem is inspired from the actual environment of a plastic container manufacturer where there are two types of setups: a mould and a color setup. During the night there are workers only for the color setup, and the mould setup cannot be performed. As a result, the color setup is unrestricted while the mould setup is restricted. The color setup involves the production and rejection of containers, and the production process requires the presence of the mould. For this reason the color setup can be performed before or after the mould setup, but the two setups cannot be performed simultaneously.

Scheduling problems with time-dependent setup times or with time-dependent processing time are equivalent, because the dependent part of the setup time, in this case the idle time, can be added either to the processing time or to the setup time. In this problem it is easier to deal with time-dependent setup times because it is the setup that causes the dependency and the idle times.

Machine scheduling problems with time-dependent setup times or processing times have received increased attention in recent years. Gupta and Gupta (1988) have introduced a scheduling model in which the processing time of a task is a polynomial function of its starting time. For this model, however, the makespan scheduling problem with polynomial time-dependent processing times is already very intricate. For this reason, most subsequent research along this line has concentrated on problems with linear or piecewise linear time-dependent processing times. This problem reflects some real-life situations in which the processing time of a task increases or decreases linearly or piecewise linearly with its starting time. One can find examples in financial management, steel production, resource allocation and national defence, where any delay in the beginning of a task may increase or decrease the duration and cost of this task. Several applications of were described by Kunnathur and Gupta (1990), Mosheiov (1994, 1996), and Sundararaghavan and Kunnathur (1994).

Alidaee and Womer (1999) and Cheng et al. (2004) have presented reviews on scheduling problems with time-dependent processing times. For the model with a single machine, the minimization of makespan and step processing times, and in particular with two steps, Mosheiov (1995) and Cheng and Ding (2001) have shown that the problem is NP-complete,

while Jeng and Lin (2004) have proposed a pseudo-polynomial time dynamic programming algorithm and a branch-and-bound algorithm. In our problem there is a forbidden part that is repeated in the same position in every period and, as a result, the number of steps in the setup time function depends on the planning horizon. In general this planning horizon consists of several periods and the function thus has multiple steps.

The (STDSP) can be seen as a time-dependent asymmetric traveling salesman problem (TDTSP) (Malandraki and Daskin, 1992) in which every node represents a job and the length of an arc is the setup time between the two corresponding jobs. The TDTSP is a TSP where the length of an arc is a function of the departure time from the origin of the arc. Malandraki and Daskin (1992) have presented a mixed integer linear programming formulation for the TDTSP and the time-dependent vehicle routing problem (TDVRP). The travel time function considered is a step function of the departure time from the first node. The authors have also studied some properties of the problem, heuristic algorithms, and a cutting plane algorithm. They have reported computational results on test problems with a travel function of two or three intervals. Later, Malandraki and Dial (1996) have presented a restricted dynamic programming heuristic as a generalization of the nearest-neighbor heuristic, and have tested it on randomly generated problems with two or three periods for each arc. Recently, the time-dependent VRP was addressed with a multiple ant colony system by Donati et al. (2003) and Haghani and Jung (2005) developed a genetic algorithm for the dynamic and time-dependent VRP.

The main contributions of this paper are threefold. First, we show how the STDSP can be modeled as a time-dependent traveling salesman problem. Second, we introduce a stronger formulation that provides strictly better linear programming lower bounds than the first formulation. Third, we introduce several families of valid inequalities that further strengthen the formulation. These inequalities are then used within a branch-and-cut algorithm which was tested on instances with up to 35 jobs.

The remainder of the paper is organized as follows. Section 2 introduces the different formulations whose strength is then compared from a theoretical point of view in Section 3. Valid inequalities are described in Section 4, followed by the branch-and-cut algorithm in Section 5. Finally, computational experiments are reported in Section 6.

2 Mathematical formulations of the problem

In this section, we introduce the notation that will be used throughout the paper, followed by three mathematical formulations of the problem. We first describe a non-linear formulation in which the setup time is given as a function. We then describe two linear formulations inspired from the TDTSP.

2.1 Notation

Let n be the number of jobs to be scheduled. For notational convenience we also introduce two dummy jobs, denoted by 0 and $n+1$, and define the set $N = \{0, 1, ..., n, n+1\}$ of all jobs. Let also $P = \{0, \ldots, n\}$ and $S = \{1, \ldots, n+1\}$ be the sets of possible predecessors and successors, respectively. Each job $j \in N \setminus \{0\}$ has a processing time p_j with $p_{n+1} = 0$

and is assumed to be ready at time t (starting time) to be processed by a single machine. Preemption is not permitted.

Let r_{ij} be the restricted setup time between jobs i and j, and u_{ij} the unrestricted setup time. Let also d denote the length of a planning period (e.g., one day) and let $[a, b] \subset [0, d]$ denote the interval during which the restricted setup cannot be performed. Without loss of generality, one can assume that $[a, b]$ is of the form $[a, d]$, i.e., the end of the forbidden interval coincides with the end of each period, because the actual position of the forbidden part during the period is irrelevant.

For $j \in N$, let t_j be a real-valued decision variable representing the completion time of job j. If $r_{ij} \geq u_{ij}$ then the total setup time s_{ij} is a function of the time at which job i is completed. Hence, $s_{ij} = f(t_i)$ where $f : \mathbb{R}^+ \to \mathbb{R}^+$ is defined as follows:

$$f(t_i) := \begin{cases} r_{ij} + u_{ij} & \text{if } t_i \in [0, a - r_{ij}], \\ r_{ij} + u_{ij} + d - a & \text{if } t_i \in (a - r_{ij}, a - u_{ij}], \\ r_{ij} + d - t_i & \text{if } t_i \in (a - u_{ij}, d - u_{ij}], \\ r_{ij} + u_{ij} & \text{if } t_i \in (d - u_{ij}, d]. \end{cases}$$

If job i terminates during the first interval, $[0, a - r_{ij}]$, it is possible to perform the restricted setup before the forbidden interval. If job i ends during the second interval, $(a - r_{ij}, a - u_{ij}]$, it is impossible to fully perform the restricted setup before a. In this case, the unrestricted setup is performed before a and the restricted setup is divided into two parts: one before and the other one after the interval $[a, d]$. As a result, $d - a$ time is lost and is added to the setup time. If job i finishes during the third interval, $(a - u_{ij}, d - u_{ij}]$, one can do the unrestricted setup, wait until the end of $[a, d]$ and then perform the restricted setup. Finally, if job i ends during the fourth interval $(d - u_{ij}, d]$ it is again possible to perform both setups without problems, as in the first interval.

This function is illustrated in Figure 1 for the interval $[0, d]$. Note that if $r_{ij} < u_{ij}$ then the second interval is unimportant. In addition, if $d - u_{ij} < a - r_{ij}$ $(d - a < u_{ij} - r_{ij})$ or if $r_{ij} = 0$ then the total setup time is constant.

2.2 Initial formulation

To model the problem, we define for each pair of jobs i, j with $i \in P$ and $j \in S$ a binary variable $x_{ij} = 1$ if and only if job j is processed immediately after job i. The problem can then be formulated as the following nonlinear mixed integer programming problem:

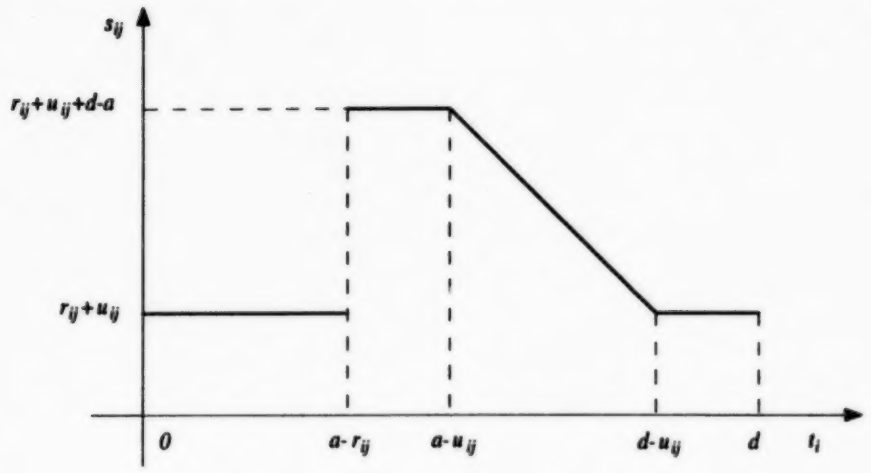

Figure 1: *Setup time from job i to j*

$$\min t_{n+1} \tag{1}$$

subject to

$$\sum_{i \in P} x_{ij} = 1 \qquad \forall j \in S \tag{2}$$

$$\sum_{j \in S} x_{ij} = 1 \qquad \forall i \in P \tag{3}$$

$$t_j - M x_{ij} \geq t_i + (s_{ij} + p_j) x_{ij} - M \qquad \forall i \in P, j \in S \tag{4}$$

$$s_{ij} = f(t_i) \qquad \forall i \in P, j \in S \tag{5}$$

$$t_i \geq t \qquad \forall i \in N \tag{6}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in P, j \in S. \tag{7}$$

The objective function (1) minimizes the completion time of the last job. Constraints (2) and (3) ensure that each job is processed exactly once. Constraints (4) ensure the consistency of the t_i variables. Constraints (5) define the setup time as a function of the time at which job i finishes. In this formulation, M is a large number used to linearize constraints (4). Any valid upper bound on the value of t_{n+1} can be used as a value for M.

2.3 Formulation as a time-dependent TSP

In formulation (1)-(7) the setup time is a function of the time at which job i terminates. To formulate the problem as a linear mixed integer program and avoid constraints (5), it is necessary to redefine the total setup time as a step function with a constant value over each

interval. This is accomplished by allowing waiting times. In this case, the second and third intervals of the setup time function become equivalent and the function is replaced with that illustrated in Figure 2.

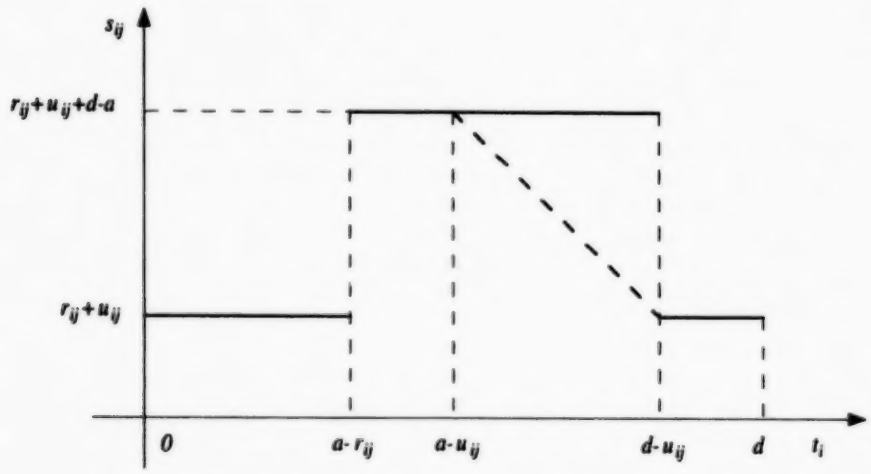

Figure 2: *Setup time between i and j*

We can formulate the problem as a TDTSP where every node represents a job and the length of every arc is the setup time between the two corresponding jobs. To this purpose, we define a complete directed graph $G = (V, A)$ where $V = N = \{0, 1, ..., n, n+1\}$ is the set of nodes and A is the set of directed arcs. The service duration in node j is represented by the processing time p_j. We also define a $n+2 \times n+2$ time-dependent matrix $C(t) = [c_{ij}(t_i)]$ representing the travel time on arc $(i, j) \in A$, where $c_{ij}(t_i)$ is a function of the time t_i at the origin node i. In this case the travel time function is a step function of the departure time from node i. This function, defined on the interval $[0, d]$, is the following:

$$c_{ij}(t_i) := \begin{cases} r_{ij} + u_{ij} & \text{if } t_i \in [0, a - r_{ij}], \\ r_{ij} + u_{ij} + d - a & \text{if } t_i \in (a - r_{ij}, d - u_{ij}), \\ r_{ij} + u_{ij} & \text{if } t_i \in [d - u_{ij}, d]. \end{cases}$$

In this way, the total period is divided into time intervals. Once the time interval in which there is the departure time from node i to j is known, the travel time between i and j is a known constant. Let K represent the set of relevant time intervals. The problem can be reformulated on an expanded network, where every arc (i, j) is replaced with $|K|$ parallel arcs from i to j, one for each time interval $k \in K$. In this new network c^k_{ij} represents the travel time from node i to j if starting at i during time interval k. We define I^k_{ij} as the upper bound of the time interval k for arc (i, j). Of course, all inbound arcs to node 0 and outbound arcs from node $n+1$ can be removed from the graph. In addition, $c^k_{i,n+1} = 0$ for each node i and for every interval k.

Let x_{ij}^k be a binary variable equal to 1 if and only if node j is visited immediately after node i and the vehicle leaves node i during interval k. Let also t_i be a continuous variable representing the departure time from node i. The TDTSP can be formulated as follows:

$$\min t_{n+1} \tag{8}$$

subject to

$$\sum_{i \in P} \sum_{k \in K} x_{ij}^k = 1 \qquad \forall j \in S \tag{9}$$

$$\sum_{j \in S} \sum_{k \in K} x_{ij}^k = 1 \qquad \forall i \in P \tag{10}$$

$$t_j - t_i - M_1 x_{ij}^k \geq (c_{ij}^k + p_j) x_{ij}^k - M_1 \qquad \forall i \in P, j \in S, k \in K \tag{11}$$

$$t_i + M_2 x_{ij}^k \leq I_{ij}^k + M_2 \qquad \forall i \in P, j \in S, k \in K \tag{12}$$

$$t_i - I_{ij}^{k-1} x_{ij}^k \geq 0 \qquad \forall i \in P, j \in S, k \in K \tag{13}$$

$$t_i \geq t \qquad \forall i \in N \tag{14}$$

$$x_{ij}^k \in \{0, 1\} \qquad \forall i \in P, j \in S, k \in K. \tag{15}$$

The objective function (8) minimizes the total time which corresponds to the arrival time at node t_{n+1}. Constraints (9) and (10) ensure that each node is visited exactly once. Constraints (11) compute the departure time at node j. These constraint are very similar to the Miller-Tucker-Zemlin (Miller et al., 1960) subtour elimination constraints for the TSP. If $x_{ij}^k = 1$ these constraints are satisfied at equality except in the case where waiting at node i decreases the objective function value. Constraints (12) and (13) ensure that the proper parallel arc k is chosen between nodes i and j according to the departure time from node i. In this formulation, the values of the constants M_1 and M_2 can be set to any upper bound on the value of t_{n+1}. Constraints (14) ensure that the departure time from every node i is greater than or equal to the starting time t. Node 0 is included in these constraints, because it may be better to wait than to start immediately (this is the case if we are at the end of the second interval).

This formulation contains $n+1$ variables t_i plus a number of x_{ij}^k variables that depends on the length of the tour (i.e., the total number of intervals). Instead of this formulation one can think of a more compact one involving a number of variables that does not depend on the total travel time. This is possible because the travel time function on arc (i, j) has the same structure in every period from $[h_i d; (h_i + 1)d]$, where h_i is a variable that represents the number of the period in which departure from node i takes place. The variables h_i are defined for $i \in N$ and we set $h_0 = 0$. In every period $[h_i d; (h_i + 1)d]$ there are three intervals and then $|K| = 3$. In Figure 3 is shown the first period of the travel time function between i and j. The new formulation is:

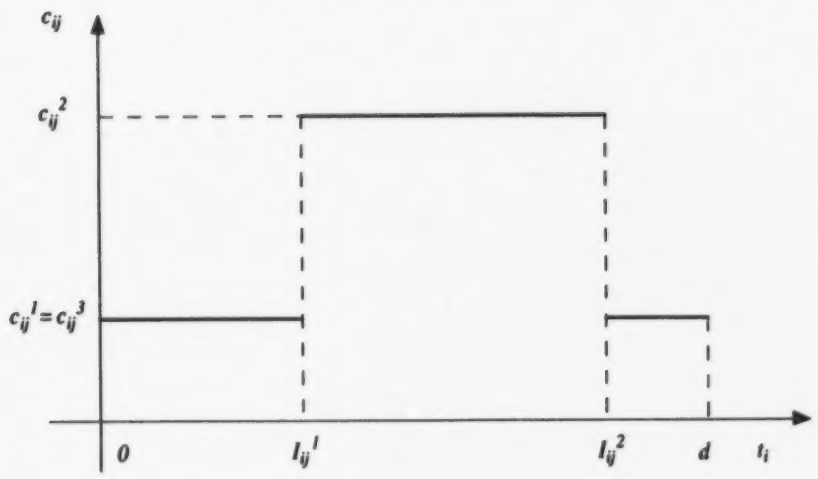

Figure 3: *Travel time function between i and j*

$$\min t_{n+1} \tag{16}$$

subject to

$$\sum_{i \in P} \sum_{k \in K} x_{ij}^k = 1 \qquad \forall j \in S \tag{17}$$

$$\sum_{j \in S} \sum_{k \in K} x_{ij}^k = 1 \qquad \forall i \in P \tag{18}$$

$$t_j - t_i - B_1 x_{ij}^k \geq (c_{ij}^k + p_j) x_{ij}^k - B_1 \qquad \forall i \in P, j \in S, k \in K \tag{19}$$

$$t_i + B_2 x_{ij}^k \leq I_{ij}^k + dh_i + B_2 \qquad \forall i \in P, j \in S, k \in K \tag{20}$$

$$t_i - I_{ij}^{k-1} x_{ij}^k - dh_i \geq 0 \qquad \forall i \in P, j \in S, k \in K \tag{21}$$

$$t_i \geq t \qquad \forall i \in N \tag{22}$$

$$h_i \in \mathbf{N} \qquad \forall i \in N \tag{23}$$

$$x_{ij}^k \in \{0, 1\} \qquad \forall i \in P, j \in S, k \in K. \tag{24}$$

This formulation is similar to the previous one with the exception that the upper bound on each interval is now defined in a different way. Here $I_{ij}^k \in [0, d]$ but t_i can take any value. The value of I_{ij}^k is thus relative to a single period.

Constraints (19) can be rewritten as follows:

$$t_j \geq t_i + B_1 \sum_{k \in K} x_{ij}^k + \sum_{k \in K} (c_{ij}^k + p_j) x_{ij}^k - B_1 \qquad \forall i \in P, j \in S.$$

The same can be done with constraints (20) and (21):

$$t_i \leq \sum_{j \in S} \sum_{k \in K} I_{ij}^k x_{ij}^k + dh_i \qquad \forall i \in P$$

$$t_i \geq \sum_{j \in S} \sum_{k \in K} I_{ij}^{k-1} x_{ij}^k + dh_i \qquad \forall i \in P.$$

This finally results in the following formulation:

$$\min t_{n+1} \tag{25}$$

subject to

$$\sum_{i \in P} \sum_{k \in K} x_{ij}^k = 1 \qquad \forall j \in S \tag{26}$$

$$\sum_{j \in S} \sum_{k \in K} x_{ij}^k = 1 \qquad \forall i \in P \tag{27}$$

$$t_j - t_i - B_1 \sum_{k \in K} x_{ij}^k \geq \sum_{k \in K} (c_{ij}^k + p_j) x_{ij}^k - B_1 \qquad \forall i \in P, j \in S \tag{28}$$

$$t_i \leq \sum_{j \in S} \sum_{k \in K} I_{ij}^k x_{ij}^k + dh_i \qquad \forall i \in P \tag{29}$$

$$t_i \geq \sum_{j \in S} \sum_{k \in K} I_{ij}^{k-1} x_{ij}^k + dh_i \qquad \forall i \in P \tag{30}$$

$$t_i \geq t \qquad \forall i \in N \tag{31}$$

$$h_i \in \mathbf{N} \qquad \forall i \in N \tag{32}$$

$$x_{ij}^k \in \{0, 1\} \qquad \forall i \in P, j \in S, k \in K. \tag{33}$$

2.4 Another formulation

The latter formulation can be rewritten in a slightly different way, giving tighter linear relaxations. To this purpose, we introduce new notation. Let t_{ij} be the departure time from node j if it is visited immediately after node i, and h_{ij} the number of the period in which node j is visited if it is visited immediately after node i. By considering the last formulation one can observe that the variables t_i and h_i are:

$$t_j = \sum_{i \in P} t_{ij} \qquad \forall j \in S$$

$$h_j = \sum_{i \in P} h_{ij} \qquad \forall j \in S.$$

and

$$t_{ij} := \begin{cases} t_j & \text{if if } j \text{ is visited after } i, \\ 0 & \text{otherwise.} \end{cases}$$

$$h_{ij} := \begin{cases} h_j & \text{if if } j \text{ is visited after } i, \\ 0 & \text{otherwise.} \end{cases}$$

We define t_0 as the departure time from node 0. Because it can be better to wait than to start immediately, we use this variable to define the real starting time. With these new variables we can reformulate the problem as follows:

$$\min \sum_{i=1}^{n} t_{i,n+1} \tag{34}$$

subject to

$$\sum_{i \in P} \sum_{k \in K} x_{ij}^k = 1 \qquad \forall j \in S \tag{35}$$

$$\sum_{j \in S} \sum_{k \in K} x_{ij}^k = 1 \qquad \forall i \in P \tag{36}$$

$$\sum_{j \in S} t_{ij} \geq \sum_{l \in P} t_{li} + \sum_{j \in S} \sum_{k \in K} (c_{ij}^k + p_j) x_{ij}^k \qquad (i = 1, 2, ..., n) \tag{37}$$

$$\sum_{j=1}^{n} t_{0j} \geq t_0 + \sum_{j=1}^{n} \sum_{k \in K} (c_{0j}^k + p_j) x_{0j}^k \tag{38}$$

$$\sum_{l \in P} t_{li} \geq \sum_{j \in S} \sum_{k \in K} I_{ij}^{k-1} x_{ij}^k + d \sum_{m \in P} h_{mi} \qquad (i = 1, 2, ..., n) \tag{39}$$

$$t_0 \geq \sum_{j=1}^{n} \sum_{k \in K} I_{0j}^{k-1} x_{0j}^k \tag{40}$$

$$\sum_{l \in S} t_{li} \leq \sum_{j \in S} \sum_{k \in K} I_{ij}^k x_{ij}^k + d \sum_{m \in S} h_{mi} \qquad (i = 1, 2, ..., n) \tag{41}$$

$$t_0 \leq \sum_{j=1}^{n} \sum_{k \in K} I_{0j}^k x_{0j}^k \tag{42}$$

$$t_{ij} \leq M_1 \sum_{k \in K} x_{ij}^k \qquad \forall i \in P, j \in S \tag{43}$$

$$h_{ij} \leq M_2 \sum_{k \in K} x_{ij}^k \qquad \forall i \in P, j \in S \tag{44}$$

$$t_0 \geq t \tag{45}$$

$$h_{ij} \in \mathbb{N} \qquad \forall i \in P, j \in S \tag{46}$$

$$x_{ij}^k \in \{0, 1\} \qquad \forall i \in P, j \in S, k \in K. \tag{47}$$

The objective function (34) minimizes the departure time from job $n + 1$ as the sum of the transition times between i and $n + 1$. Only one $t_{i,n+1}$ variable can be positive, because there is only one variable $x_{i,n+1}^k$ equal to one and by constraints (43) there is only one i such that $t_{i,n+1} > 0$. Constraints (35) and (36) ensure that each node is visited exactly once.

Constraints (37) compute the departure time from the node j visited after i. For every node i there is only one $t_{ij} > 0$ and only one $t_{li} > 0$, and $t_{ij} \geq t_{li}$. To the departure time from job i it is necessary to add the transition time between i and j. Constraint (38) computes the departure time from a node j that is visited immediately after node 0. Constraints (39) and (41) ensure that the interval k is chosen between i and j according to the departure time from node i. Constraints (40) and (42) are like the previous constraints but if node j is visited immediately after 0. Constraints (43) and (44) ensure that $t_{ij} \geq 0$ and $h_{ij} \geq 0$ if and only if one of $x_{ij}^k = 1$ and $t_{ij} = 0$, and $h_{ij} = 0$ otherwise. In the formulation there are two large numbers M_1 and M_2. Any upper bound on the value of $\sum_{i=1}^{n} t_{i,n+1}$ and $\sum_{i=1}^{n} t_{i,n+1}/d$ can be used as a value for M_1 and M_2, respectively.

3 Comparison of the formulations

In this section, we prove two results that compare the strength of the different formulations introduced in the previous section.

Proposition 1 *The value of the lower bound obtained by solving the LP relaxation of formulation (8)-(15) is equal to the value of the lower bound obtained by solving the LP relaxation of formulation (25)-(33).*

Proof. We first show that the value of the objective function of an optimal solution for the LP relaxation of formulation (8)-(15) is equal to t. Because of constraints (14) the value of the objective function is at least t. Now, $x_{ij}^1 = 1/n \ \forall i \in P, j \in S$; $x_{ij}^k = 0 \ \forall i \in P, j \in S$, $k \neq 1$; $t_{n+1} = t$; and $t_i = 0$ $(i = 1, 2, ..., n)$, is a solution to the LP relaxation of formulation (8)-(15). For this solution the value of the objective function is equal to t, and is thus optimal.

We now show that the value of the objective function of an optimal solution the LP relaxation of formulation (25)-(33) is equal to t. Because of constraints (31) the value of the objective function is at least t. Now, $x_{ij}^1 = 1/n \ \forall i \in P, j \in S$; $x_{ij}^2 = x_{ij}^3 = 0 \ \forall i \in P, j \in S$; $t_{n+1} = t$; $t_i = 0$ $(i = 1, 2, ..., n)$; and $h_i = 0 \ \forall i \in N$, is a solution to the LP relaxation of formulation (25)-(33). For this solution the value of the objective function is equal to t, and is thus optimal. Hence, the LP relaxations of the two formulations have the same value. □

Proposition 2 *The value of the lower bound obtained by solving the LP relaxation of formulation (25)-(33) is less than the value of the lower bound obtained by solving the LP relaxation of formulation (34)-(47).*

Proof. We know from Proposition 1 that the lower bound obtained by solving the LP relaxation of formulation (25)-(33) is equal to t. We now show that the value of the objective function of an optimal solution to the LP relaxation of (34)-(47) is greater than t. By summing all constraints (37) we obtain:

$$\sum_{i=1}^{n} \sum_{j \in S} t_{ij} \geq \sum_{i=1}^{n} \sum_{l \in P} t_{li} + \sum_{i=1}^{n} \sum_{j \in S} \sum_{k \in K} (c_{ij}^k + p_j) x_{ij}^k$$

and

$$\sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij} + \sum_{i=1}^{n} t_{i,n+1} \geq \sum_{i=1}^{n}\sum_{l=1}^{n} t_{li} + \sum_{i=1}^{n} t_{0i} + \sum_{i=1}^{n}\sum_{j\in S}\sum_{k\in K}(c_{ij}^{k} + p_j)x_{ij}^{k}$$

$$\sum_{i=1}^{n} t_{i,n+1} \geq \sum_{i=1}^{n} t_{0i} + \sum_{i=1}^{n}\sum_{j\in S}\sum_{k\in K}(c_{ij}^{k} + p_j)x_{ij}^{k}$$

and then from constraints (38) it is possible to write

$$\sum_{i=1}^{n} t_{i,n+1} \geq \sum_{i=1}^{n} t_{0i} + \sum_{i=1}^{n}\sum_{j\in S}\sum_{k\in K}(c_{ij}^{k} + p_j)x_{ij}^{k} \geq t_0 + \sum_{i=0}^{n}\sum_{j\in S}\sum_{k\in K}(c_{ij}^{k} + p_j)x_{ij}^{k}$$

and, from constraint (45),

$$\sum_{i=1}^{n} t_{i,n+1} \geq t + \sum_{i=0}^{n}\sum_{j\in S}\sum_{k\in K}(c_{ij}^{k} + p_j)x_{ij}^{k}.$$

Since $\sum_{i=0}^{n}\sum_{j\in S}\sum_{k\in K}(c_{ij}^{k} + p_j)x_{ij}^{k} > 0$, one may conclude that

$$\sum_{i=1}^{n} t_{i,n+1} > t. \quad \square$$

4 Valid inequalities

We now present some families of valid inequalities that can be added to formulation (34)-(47) in order to strengthen its linear relaxation. These inequalities are either taken from the literature and adapted to the TDTSP, or are specific to this problem.

4.1 Subtour elimination constraints

All valid inequalities for the asymmetric TSP (ATSP) are also valid for the STDSP. To this end, it is necessary to take into account the existence of the three intervals and to replace all the x_{ij} variables of the ATSP with $\sum_{k\in K} x_{ij}^{k}$.

In particular we consider the asymmetric inequalities introduced by Grötschel and Padberg (1985) and known in literature as D_k^+ and D_k^- inequalities. Here, for notational convenience we call them D_l^+ and D_l^-.

Proposition 3 *Let* $\{i_1, i_2, ..., i_l\} \subset N \setminus \{0, n+1\}$, $3 \leq l \leq n-1$. *The* D_l^+ *inequality*

$$\sum_{k\in K} x_{i_1 i_l}^{k} + \sum_{k\in K}\sum_{h=2}^{l} x_{i_h i_{h-1}}^{k} + 2\sum_{k\in K}\sum_{h=2}^{l-1} x_{i_1 i_h}^{k} + \sum_{k\in K}\sum_{h=3}^{l-1} x(\{i_2, ..., i_{h-1}\}, i_h)^{k} \leq l-1 \qquad (48)$$

and the D_l^- inequality

$$\sum_{k\in K} x^k_{i_l i_1} + \sum_{k\in K}\sum_{h=2}^{l} x^k_{i_{h-1}i_h} + 2\sum_{k\in K}\sum_{h=2}^{l-1} x^k_{i_h i_1} + \sum_{k\in K}\sum_{h=3}^{l-1} x(i_h, \{i_2, ..., i_{h-1}\})^k \leq l-1 \qquad (49)$$

are valid for the problem.

Proof. These inequalities with x_{ij} variables are valid for the ATSP, as proved by Grötschel and Padberg (1985). Our problem is formulated as an ATSP where every arc between two nodes is replaced by tree parallel arcs and the x_{ij} variables are replaced by $\sum_{k\in K} x^k_{ij}$. For this reason every inequality for the ATSP is also valid for this problem.□

4.2 Bounds on periods

Some valid inequalities involving the h_{ij} variables can be obtained from constraints (37) and (38).

Proposition 4 *The following inequalities are valid for the STDSP:*

$$\sum_{j\in S} h_{ij} \geq \sum_{l\in P} h_{li} + \sum_{j\in S}\sum_{k\in K} \lfloor \frac{c^k_{ij} + p_j}{d} \rfloor x^k_{ij} \qquad (i = 1, 2, ..., n) \qquad (50)$$

$$\sum_{j=1}^{n} h_{0j} \geq \sum_{j\in S}\sum_{k\in K} \lfloor \frac{t + c^k_{0j} + p_j}{d} \rfloor x^k_{0j}. \qquad (51)$$

Proof. For node i, on has

$$\sum_{j\in S} h_{ij} \geq \sum_{l\in P} h_{li}$$

because h_{ij} is the number of the period in which job j is completed if it is performed after job i. Then, $\lfloor \frac{c^k_{ij}+p_j}{d} \rfloor$ is the travel time between i and j expressed in periods. The period of departure from node j is larger than the period of departure from node i plus the travel time between i and j expressed in periods. For inequality (51), one can make the same argument.□

4.3 Bounds on time and period

Proposition 5 *The following inequalities are valid for the STDSP:*

$$t_{ij} - dh_{ij} \leq d\sum_{k\in K} x^k_{ij} \qquad \forall i \in P, j \in S. \qquad (52)$$

Proof. If node j is not visited after i then $\sum_{k\in K} x^k_{ij} = 0$, $t_{ij} = 0$, $h_{ij} = 0$ and the inequality is valid. If node j is visited after i then one of x^k_{ij} is equal to 1. For every pair of nodes i and j, $t_{ij} \geq dh_{ij}$ and $t_{ij} - dh_{ij} \leq d$,because h_{ij} can be considered as the integer part of the division of t_{ij} by d. □

4.4 Bounds on intervals

This group of inequalities on intervals is defined for nodes whose travel time function is made up of two or three intervals. We do not consider travel time functions with only the first interval, because for every pair of nodes i and j our aim is to force some variables depending on the interval to be equal to zero (for instance $x^2_{ij} = 0$). Hence, for every inequality in this group, $i \in P, j \in S \setminus \{n+1\}, r_{i,j} \neq 0,\ d - a \geq u_{ij} - r_{ij}$. Before introducing the first group of inequalities, we define for notational convenience a function

$$g^1 : [0, I^1_{ij}] \to [0, d] \qquad\qquad i \in P, j \in S \setminus \{n+1\}$$
$$x \to (x + c^1_{ij} + p_j) \mod d$$

and some node subsets:

$$A = \{l \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [0, I^1_{ij}]\ \ g^1(x) \leq I^1_{jl}\}$$
$$B = \{m \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [0, I^1_{ij}]\ \ I^1_{jm} < g^1(x) < a - u_{jm}\}$$
$$C = \{p \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [0, I^1_{ij}]\ \ g^1(x) \geq a - u_{jp}\}$$
$$D = \{q \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \forall\, x \in [0, I^1_{ij}]\ \ g^1(x) > I^1_{jq}\}$$
$$E = \{r \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \ g^1(0) \geq a - u_{jr}\ \wedge g^1(I^1_{ij}) \leq I^1_{jr}\}$$
$$F = \{s \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \forall\, x \in [0, I^1_{ij}]\ \ g^1(x) < a - u_{js}\}.$$

Function g^1 relates each point in the interval $[0, I^1_{ij}]$ to another point in interval $[0, d]$, using the travel time between i and j. This function is defined for every node i in P and j in $S \setminus \{n+1\}$. We can relate all the domain interval to another interval in $[0, d]$. If node l is such that the range of the function is an interval in $[0, I^1_{jl}]$ then we put node l in set A. In Figure 4 there is the description of this case, considering the first period of the function ($h_{ij} = 0$). The two broken arrows represent $g^1(0)$ and $g^1(I^1_{ij})$, respectively. If the range of the function is an interval in $(I^1_{jm}, a - u_{jm})$ for some nodes m we put these nodes in set B. In set C are all the nodes p such that the range of the function is an interval in $[a - u_{jp}, d]$. If the range is an interval in (I^1_{jq}, d) the nodes not already inserted in the other sets A, B and C are inserted in set D. If the range is made up of two different intervals one in $[0, I^1_{jr}]$ and another one in $[a - u_{jr}, d]$, these nodes not already inserted are inserted in E and in F are the nodes s not already inserted where the range is in $[0, a - u_{js})$.

Proposition 6 *For all $i \in P$ and $j \in S \setminus \{n+1\}$, the following inequality is valid for the STDSP:*

$$x^1_{ij} + \sum_{l \in A}(x^2_{jl} + x^3_{jl}) + \sum_{m \in B}(x^1_{jm} + x^3_{jm}) + \sum_{p \in C}(x^1_{jp} + x^2_{jp}) + \sum_{q \in D} x^1_{jq} + \sum_{r \in E} x^2_{jr} + \sum_{s \in F} x^3_{js} \leq 1. \quad (53)$$

Proof. With this group of constraints we want to put in relation two nodes i and j with any possible successor node of j. To do this, we use the information about the travel time between i and j and the interval in which there is the departure time from node j in the travel function between j and any possible node successor of j. If x^1_{ij} is equal to one then

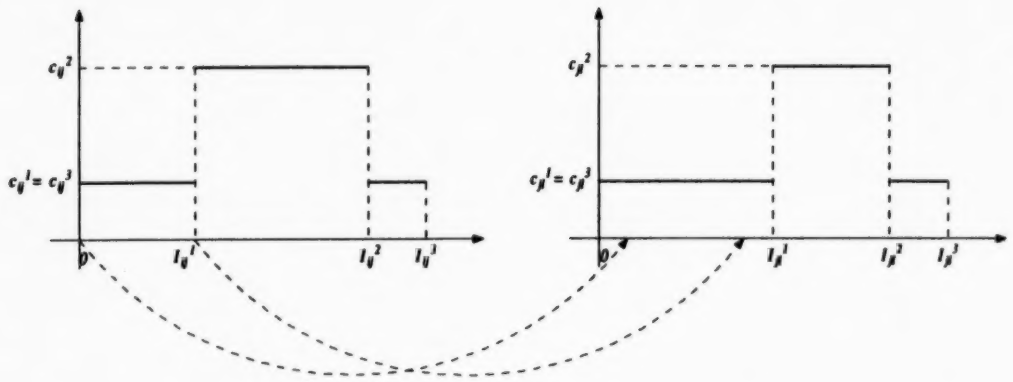

Figure 4: *Relation between the travel time functions from i to j and from j to l*

node j is visited after i and the departure time from node i is in the first interval. In this way we know the travel time from i to j equal to $c_{ij}^1 + p_j$ and it is possible to calculate the departure time from node j visited after i (t_{ij}) (Figure 4). Knowing the node successor of j and the travel time function between j and the successor we can know in which interval is t_{ij}. Let us consider all nodes l in set A. For them if the departure time from node i to node j is in the first interval ($x_{ij}^1 = 1$) then the departure time from node j to l is in the first interval. For this reason we can say that x_{jl}^2 and x_{jl}^3 are equal to zero. For nodes m in set B and p in C we can say that x_{jm}^1, x_{jm}^3 and x_{jp}^1, x_{jp}^2 are equal to zero respectively. For nodes q in D we can only assert that x_{jq}^1, because the range of the function is in two of the three intervals. For nodes r in set E and s in F we can say that x_{jr}^2 and x_{js}^3 are equal to zero, respectively.

We know by construction that sets A, B, C, D, E and F are disjoint. From the formulation we know that $\sum_{l \in A}(x_{jl}^2 + x_{jl}^3) + \sum_{m \in B}(x_{jm}^1 + x_{jm}^3) + \sum_{p \in C}(x_{jp}^1 + x_{jp}^2) + \sum_{q \in D} x_{jq}^1 + \sum_{r \in E} x_{jr}^2 + \sum_{s \in F} x_{js}^3 \leq 1$ because from j one can visit only one other vertex. Now we prove that either x_{ij}^1 or one of the variables of the previous inequality can be equal to one. We proove the inequality by contradiction. Let x_{ij}^1 and x_{jl}^2 with $l \in A$ be equal to one. Because $l \in A$ then if we start from the first interval between i and j we arrived in the first interval between j and l. But x_{ij}^1 and x_{jl}^2 are both equal to one and it means that we started from the first interval and we arrived to the second interval between j and l. This is not possible because we started from one interval and we arrive in two different intervals. For this reason either x_{ij}^1 or x_{jl}^2 is equal to one. The same can be said for the other variables of the inequality. □

For the next group of inequalities we define a function

$$g^2 : [l_{ij}^1, a - u_{ij}] \rightarrow [0, d] \qquad i \in P, j \in S \setminus \{n+1\}$$
$$x \rightarrow (x + c_{ij}^2 + p_j) \mod d$$

and some new node subsets as follows:

$$A = \{l \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [I^1_{ij}, a-u_{ij}]\ \ g^2(x) \le I^1_{jl}\}$$
$$B = \{m \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [I^1_{ij}, a-u_{ij}]\ \ I^1_{jm} < g^2(x) < a-u_{jm}\}$$
$$C = \{p \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [I^1_{ij}, a-u_{ij}]\ \ g^2(x) \ge a-u_{jp}\}$$
$$D = \{q \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \forall\, x \in [I^1_{ij}, a-u_{ij}]\ \ g^2(x) > I^1_{jq}\}$$
$$E = \{r \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \ g^2(I^1_{ij}) \ge a-u_{jr}\ \wedge g^2(a-u_{ij}) \le I^1_{jr}\}$$
$$F = \{s \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \forall\, x \in [I^1_{ij}, a-u_{ij}]\ \ g^2(x) < a-u_{js}\}.$$

Proposition 7 *For all $i \in P$ and $j \in S \setminus \{n+1\}$, the following inequality is valid for STDSP:*

$$x^2_{ij} + \sum_{l \in A}(x^2_{jl} + x^3_{jl}) + \sum_{m \in B}(x^1_{jm} + x^3_{jm}) + \sum_{p \in C}(x^1_{jp} + x^2_{jp}) + \sum_{q \in D} x^1_{jq} + \sum_{r \in E} x^2_{jr} + \sum_{s \in F} x^3_{js} \le 1. \quad (54)$$

The construction of the sets and the proof for this group of inequalities is very similar to the proof of Proposition 6.

For the next group of inequalities we define another function

$$g^3 :\ [a-u_{ij}, d] \to [0,d] \qquad \forall\, i \in P, j \in S \setminus \{n+1\}$$
$$g^3(x) := \begin{cases} (d + r_{ij} + p_j) \mod d & \text{if } x \in [a-u_{ij}, d-u_{ij}), \\ (x + c^3_{ij} + p_j) \mod d & \text{if } x \in [d-u_{ij}, d] \end{cases}$$

and other sets:

$$A = \{l \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [a-u_{ij}, d]\ \ g^3(x) \le I^1_{jl}\}$$
$$B = \{m \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [a-u_{ij}, d]\ \ I^1_{jm} < g^3(x) < a-u_{jm}\}$$
$$C = \{p \in S \setminus \{i,j,n+1\} |\ \forall\, x \in [a-u_{ij}, d]\ \ g^3(x) \ge a-u_{jp}\}$$
$$D = \{q \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \forall\, x \in [a-u_{ij}, d]\ \ g^3(x) > I^1_{jq}\}$$
$$E = \{r \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \ g^3(a-u_{ij}) \ge a-u_{jr}\ \wedge g^3(d) \le I^1_{jr}\}$$
$$F = \{s \in S \setminus \{i,j,n+1\} \setminus (A \cup B \cup C) |\ \forall\, x \in [a-u_{ij}, d]\ \ g^3(x) < a-u_{js}\}.$$

Function $g^3(x)$ is defined in a slightly different way. We consider one part of the second interval and all of the third as the domain. Regarding the third interval, $g^3(x)$ has the same characteristics as $g^1(x)$ and $g^2(x)$. If the departure time from node i is in the interval $[a-u_{ij}, d-u_{ij}]$, we can see by the formulation that it is better to wait until the beginning of the third interval $(d-u_{ij})$ and then sum $c^3_{ij} + p_j$ because this decreases the objective function. Then, $d - u_{ij} + c^3_{ij} + p_j = d + r_{ij} + p_j$.

Proposition 8 *For all $i \in P$ and $j \in S \setminus \{n+1\}$, the following inequality is valid for the STDSP:*

$$x^3_{ij} + \sum_{l \in A}(x^2_{jl} + x^3_{jl}) + \sum_{m \in B}(x^1_{jm} + x^3_{jm}) + \sum_{p \in C}(x^1_{jp} + x^2_{jp}) + \sum_{q \in D} x^1_{jq} + \sum_{r \in E} x^2_{jr} + \sum_{s \in F} x^3_{js} \le 1. \quad (55)$$

As before, the construction of the sets and the proof for this group of inequalities on the third interval is very similar to the proof of Proposition 6.

Before introducing the next groups of inequalities for we define three other functions:

$$\begin{aligned}
&g_m^4 : [0,d] \to [0,d] && i \in S \setminus \{n+1\}, m \in P \setminus \{i\}\\
&\quad x \to (x - c_{mi}^1 - p_i) \mod d,\\
&g_m^5 : [0,d] \to [0,d] && i \in S \setminus \{n+1\}, m \in P \setminus \{i\}\\
&\quad x \to (x - c_{mi}^2 - p_i) \mod d,\\
&g_m^6 : [0,d] \to [0,d] && i \in S \setminus \{n+1\}, m \in P \setminus \{i\}\\
&\quad x \to (x - c_{mi}^3 - p_i) \mod d,
\end{aligned}$$

and some node subsets:

$$\begin{aligned}
A &= \{l \in P \setminus \{i,j\} |\ \forall\, x \in [0, I_{ij}^1]\ \ g_l^4(x) \le I_{li}^1\}\\
B &= \{m \in P \setminus \{i,j\} |\ \forall\, x \in [0, I_{ij}^1]\ \ I_{mi}^1 < g_m^5(x) < a - u_{mi}\}\\
C &= \{p \in P \setminus \{i,j\} |\ \forall\, x \in [0, I_{ij}^1]\ \ g_p^6(x) \ge a - u_{pi}\}\\
D &= \{s \in S \setminus \{i,j\} |\ \ I_{is}^1 \le I_{ij}^1\}.
\end{aligned}$$

The functions g_m^4, g_m^5 and g_m^6 are defined for every node $m \in P \setminus \{i\}$ and they relate each point in the interval $[0,d]$ to another point in $[0,d]$ using the travel time between node i and this node m. We know node i and x represent the departure time from node i to another node. With function g_m^4 we want to calculate the departure time from m visited directly before i if we started from m during the first interval and the travel time was c_{mi}^1. Then, if we started from m during the first interval we subtract the travel time c_{mi}^1. With g_m^5 we suppose that we started from m during the second interval, and we subtract c_{mi}^2. With g_m^6, we assume to start from the third interval.

To construct the sets we consider only $x \in [0, I_{ij}^1]$, because in the next inequalities of Proposition 9 we consider x_{ij}^1 and we suppose that the departure time from node i to j is in the first interval in the travel time function between i and j. Then, if x represents the departure time from node i, $x \in [0, I_{ij}^1]$. For each predecessor l of i we know the travel time function $(c_{li}^k + p_i)$. In the set A are all the nodes l such that if the departure time from node i is in the first interval between i and j then this means that we started from a node l whose departure time is in the first interval between l and i (Figure 5). The two broken arrows represent $g_l^4(0)$ and $g_l^4(I_{ij}^1)$. Let us consider node $m \in B$. If node m is such that if we are in the first interval between i and j this means that we started from node m during the second interval. In this case we use the function g_m^5. In C are all the nodes p such that if we are in the first interval between i and j this means that we started from node p during the third interval. To construct the sets we use each time a different function because, for instance with set A, if we want to subtract $c_{li}^1 + p_i$ it is necessary to start from the first interval between l and i, otherwise we cannot use c_{li}^1. For this reason if l is such that $g_l^4(x) \le I_{li}^1$, where $g_l^4(x)$ represents the departure time from node l if we started from the first interval, we are sure that l is a node whose departure time is in the first interval. If, for instance, $g_l^4(x) > I_{li}^1$ we cannot subtract c_{li}^1 because the departure time from node l is in the

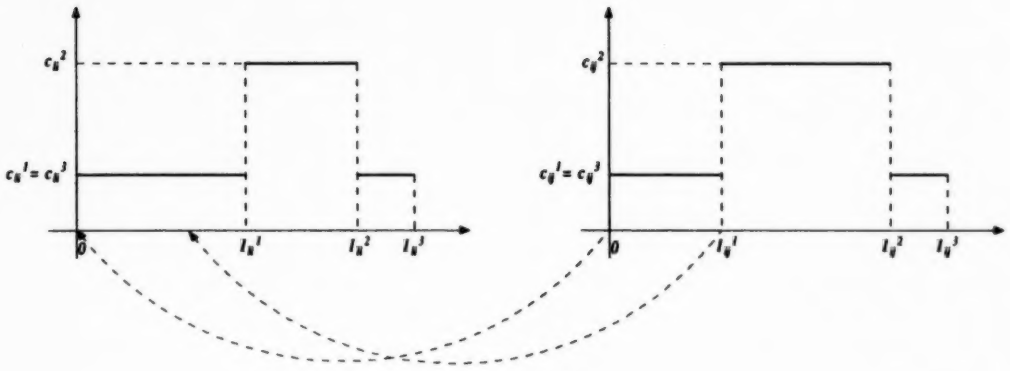

Figure 5: *Relation between the travel time functions from l to i and from i to j*

second interval. For this reason we define also $g_l^5(x)$ and $g_l^6(x)$ to consider the other starting intervals. In set D are all the nodes s whose upper bound of the fist interval from i to s is less than or equal to I_{ij}^1. If we construct sets A, B and C instead of in relation between the first interval between i and j, in relation between the first interval between i and s we have that $A \subseteq A'$, $B \subseteq B'$ and $C \subseteq C'$. Then nodes s in D have the same characteristics as j in terms of relation with i.

Proposition 9 *For all $i, j \in S \setminus \{n+1\}$, the following inequality is valid for the STDSP:*

$$x_{ij}^1 + \sum_{l \in A}(x_{li}^2 + x_{li}^3) + \sum_{m \in B}(x_{mi}^1 + x_{mi}^3) + \sum_{p \in C}(x_{pi}^1 + x_{pi}^2) + \sum_{s \in D} x_{is}^1 \leq 1. \tag{56}$$

Proof. With this group of constraints we want to put in relation two nodes i and j with any possible predecessor node of i. To do this we use the information about the travel time between any possible predecessor of i and j and the interval in which there is the departure time from node i to j. If x_{ij}^1 is equal to one then node j is visited after i and the departure time from node i is in the first interval of the travel function between i and j. We want to know the possible predecessors of i. We can arrive at i from a node whose departure time is in the first, in the second or in the third interval. Let us consider all nodes l in A. For each of them if the departure time from node i is in the first interval then we know that we started from l in the first interval. Then we can say that x_{li}^2 and x_{li}^3 are equal to zero. For nodes m in B and p in C we can say that x_{mi}^1, x_{mi}^3 and x_{pi}^1, x_{pi}^2 are equal to zero, because we started from m during the second interval and from p during the third interval. In the set D are all the nodes s that have the same characteristics as j in terms of travel time function between i and its predecessors. In fact all the conditions satisfied for every node l in A in relation with j are also satisfied for every node s in D.

We know by construction that the sets A, B, C are disjoint. We know from the formulation that $\sum_{l\in A}(x^2_{li}+x^3_{li})+\sum_{m\in B}(x^1_{mi}+x^3_{mi})+\sum_{p\in C}(x^1_{pi}+x^2_{pi})\leq 1$ because one can arrive in i only from one node and $x^1_{ij}+\sum_{s\in D}x^1_{is}\leq 1$ because after i we can directly visit only another node. Now we show that only one variables of the first inequality or one of the second is equal to one. We proof the inequality by contradiction. Let x^2_{li} with $l\in A$ and x^1_{ij} be equal to one. Then, this means that because $l\in A$ if we start from the first interval between l and i we arrive in the first interval between i and j. But if x^2_{li} and x^1_{ij} are equal to one then we started from the second interval between l and i and we arrived in the first interval between i and j. This is not possible because we started from two different intervals and we arrived in the same. For this reason either x^2_{li} with $l\in A$ or x^1_{ij} is equal to one. The same can be said for the other variables of the first inequality written above. We can make the same argument for x^1_{is} with $s\in D$ with the variables of the first inequality, because $I^1_{is}\leq I^1_{ij}$ and then j and s have the same characteristics. □

For the next group of inequalities we introduce some sets of nodes:

$$A=\{l\in P\setminus\{i,j\}|\ \forall\, x\in[I^1_{ij},a-u_{ij}]\ \ g^4_l(x)\leq I^1_{li}\}$$
$$B=\{m\in P\setminus\{i,j\}|\ \forall\, x\in[I^1_{ij},a-u_{ij}]\ \ I^1_{mi}<g^5_m(x)<a-u_{mi}\}$$
$$C=\{p\in P\setminus\{i,j\}|\ \forall\, x\in[I^1_{ij},a-u_{ij}]\ \ g^6_p(x)\geq a-u_{pi}\}$$
$$D=\{s\in S\setminus\{i,j\}|I^1_{ij}\ <I^1_{is}\wedge a-u_{is}<a-u_{ij}\}.$$

The construction of these sets is similar to the construction of the others defined above for Proposition 9. In this case we consider only $x\in[I^1_{ij},a-u_{ij}]$ and then we suppose that the departure time from node i is in the second interval.

Proposition 10 *For all $i,j\in S\setminus\{n+1\}$, the following inequality is valid for the STDSP:*

$$x^2_{ij}+\sum_{l\in A}(x^2_{li}+x^3_{li})+\sum_{m\in B}(x^1_{mi}+x^3_{mi})+\sum_{p\in C}(x^1_{pi}+x^2_{pi})+\sum_{s\in D}x^2_{is}\leq 1. \tag{57}$$

The proof for this group of inequalities is very similar to the proof of Proposition 9.

As before we define some sets of nodes:

$$A=\{l\in P\setminus\{i,j\}|\ \forall\, x\in[a-u_{ij},I^3_{ij}]\ \ g^4_l(x)\leq I^1_{li}\}$$
$$B=\{m\in P\setminus\{i,j\}|\ \forall\, x\in[a-u_{ij},I^3_{ij}]\ \ I^1_{mi}<g^5_m(x)<a-u_{mi}\}$$
$$C=\{p\in P\setminus\{i,j\}|\ \forall\, x\in[a-u_{ij},I^3_{ij}]\ \ g^6_p(x)\geq a-u_{pi}\}$$
$$D=\{s\in S\setminus\{i,j\}|a-u_{ij}\geq a-u_{is}\}$$

These sets are define as the other written above with $x\in[a-u_{ij},I^3_{ij}]$.

Proposition 11 *For all $i,j\in S\setminus\{n+1\}$, the following inequality is valid for the STDSP:*

$$x^3_{ij}+\sum_{l\in A}(x^2_{li}+x^3_{li})+\sum_{m\in B}(x^1_{mi}+x^3_{mi})+\sum_{p\in C}(x^1_{pi}+x^2_{pi})+\sum_{s\in D}x^3_{is}\leq 1 \tag{58}$$

The proof for this group of inequalities on the third interval is very similar to the proof of Proposition 9.

4.5 Bounds on starting time and period

With this group of inequalities we want to set some bounds on t_{ij} and h_{ij} variables. A data of the problem is the starting time t. Then, it is possible to set a bound on t_{ij} and h_{ij} variables knowing for each node j the interval in which takes place the starting time t. Before introducing the bounds we define st_j equal to the departure time from node j if it is visited directly after node 0:

$$st_j := \begin{cases} t + c^1_{0j} + p_j & \text{if } t \in [0, I^1_{0j}], \\ t + c^2_{0j} + p_j & \text{if } t \in (I^1_{0j}, a - u_{0j}], \\ d + r_{0j} + p_j & \text{if } t \in (a - u_{0j}, I^2_{0j}], \\ t + c^3_{0j} + p_j & \text{if } t \in (I^2_{0j}, d]. \end{cases}$$

In the third case of this definition we set the departure time from node j equal to $d + r_{0j} + p_j$ if t is in one part of the second interval ($(a - u_{0j}, I^2_{0j}]$). If t is in this interval, by the formulation, we know that it is better to wait until the beginning of the third interval instead of starting in t and summing c^2_{0j} because this decreases the objective function. Then we wait until the beginning of the third interval $(d - u_{0j})$, we sum c^3_{0j} and then the result is $d - u_{0j} + c^3_{0j} + p_j = d - r_{0j} + p_j$.

Proposition 12 *The following inequalities are valid for the STDSP:*

$$\sum_{i=0}^{n} t_{ij} \geq \min\{st_j, \min_{q \in S \setminus \{j,n+1\}} st_q + \min_{m \in S \setminus \{j,n+1\}} c^1_{mj} + p_j\} \tag{59}$$

$$\sum_{i=0}^{n} h_{ij} \geq \min\{\lfloor \frac{st_j}{d} \rfloor, \lfloor \frac{\min_{q \in S \setminus \{j,n+1\}} st_q + \min_{q \in P \setminus \{j\}} c^1_{mj} + p_j}{d} \rfloor\}. \tag{60}$$

Proof. With the first inequality (59) we want to set a bound on the starting time from node j. If node j is visited directly after node 0 then the departure time from node j is equal to st_j. If node j is not visited directly after node 0 then there is at least another node between 0 and j. From node 0 we choose the smallest departure time to exit from 0 ($\min_{q \in S \setminus \{j,n+1\}} st_q$) and to go to j we choose the shortest travel time between one node m to j ($\min_{m \in P \setminus \{j\}} c^1_{mj}$). Then we sum the two minimum values and the service duration of node j to have a bound. With the last inequality (60) we set a bound on period of the departure time from node j. We know that h_{ij} is a variable that is as the integer part of the division of t_{ij} by d to express the time in periods.

5 Branch-and-cut algorithm

Branch-and-cut is a popular technique which has been applied successfully to several hard combinatorial problems (see, e.g., Ascheuer et al. (2001)).

After applying the preprocessing steps presented in Section 5.1 and generating an initial set of inequalities, our algorithm first solves the LP relaxation of the problem. If the solution to the LP relaxation is integer, an optimal solution has been identified. Otherwise, an enumeration tree is constructed and an attempt is made to generate violated valid inequalities at each node of this tree. Because all inequalities described in Section 4 are valid for the original formulation, the inequalities added at any node of the tree are also valid for all other nodes. Hence, whenever the LP bound is evaluated at a given node of the tree, the linear program incorporates all cuts generated so far. Before starting the branch-and-cut process, an upper bound is computed with a tabu search heuristic described by Stecco (2006). This upper bound is used to prune the enumeration tree whenever the solution value at a given node exceeds that of the upper bound.

5.1 Preprocessing

Formulation (34)-(47) is defined on a complete graph G. However, because of intervals, some arcs can in fact be removed from the graph as they cannot belong to a feasible solution. As a result, some variables can be fixed:

- $x^2_{0j} = x^3_{0j} = 0 \quad \forall j \in S \setminus \{n+1\}$ if $t \leq I^1_{0j}$
- $x^1_{0j} = x^3_{0j} = 0 \quad \forall j \in S \setminus \{n+1\}$ if $I^1_{0j} < t < a - u_{0j}$
- $x^1_{0j} = x^2_{0j} = 0 \quad \forall j \in S \setminus \{n+1\}$ if $a - u_{0j} \leq t < I^3_{0j}$.

In the fist case, if the starting time t is in the first interval in the travel time function between 0 and j we can set x^2_{0j} and x^3_{0j} equal to zero, because if node j is visited directly after node 0 then it has to start in the first interval. The same can be said for the other two cases. It is also possible to set $x^2_{ij} = 0$ if $u_{ij} > r_{ij}$ because in this case the second interval is irrelevant.

In the preprocessing phase we also add to the formulation the bounds on starting time and period presented in subsection (4.5).

5.2 Cut generation

In our branch-and-cut algorithm we add only the inequalities on intervals defined in Section 4.4. Indeed, our initial experiments have shown that the other inequalities and the subtour elimination constraints do not have a significant impact on the performance of the algorithm. Each family of inequalities is polynomial in size and we thus check for violations by complete enumeration at each node of the branch-and-cut tree.

6 Computational Experiments

The branch-and-cut algorithm was implemented in C++ by using ILOG Concert 1.3 and CPLEX 9.1.3. It was run on a 3.0 GHz Pentium 4 computer with 1 Gb of memory. Branching variable selection was performed according to the maximum integer infeasibility rule, i.e., the branching variable is the one whose value is the farthest from the nearest integer. However,

branching priorities were used so as to give higher priority to the x_{ij}^k variables and lower priority to the h_{ij} variables.

The algorithm was tested on four sets of randomly generated instances. In all instances, the length of a period (d) is equal to 1440, i.e., the length of one day expressed in minutes. In Table 1 we indicate the characteristics of the four data sets. The second column (a) shows the time (in minutes) corresponding to the beginning of the forbidden part in the first period of the planning horizon (this period repeats in the same position in every day). The third, fourth and fifth columns indicate the maximum values of the setup times (r_{ij} and u_{ij}) and the processing time (p_j). All data were generated from a uniform distribution on the interval from 0 to the maximum value.

	a	r_{ij}	u_{ij}	p_j
a-1/a-10	1320	60	60	240
b-1/b-10	1200	120	120	720
c-1/c-10	960	240	240	1440
d-1/d-10	960	480	480	1440

Table 1: Characteristics of the four data sets

Tables 2, 3, 4 and 5 explain the characteristics of these four groups of instances in terms of the number n of jobs to schedule, number of constraints and variables and value of the LP relaxation at the root node of the branch-and-cut tree. The values are computed with or without the application of the preprocessing phase described in Subsection 5.1. It is worth pointing out that when the preprocessing is applied, the number of constraints actually increases because of the addition of the bounds on starting time and period. However, the LP relaxation value does not change. The tables also report the value of an upper bound obtained by running for 1000 iterations a simple tabu search heuristic which is described in more detail by Stecco (2006). This heuristic uses three operators: relocate a job, swap two jobs and relocate a couple of jobs. The last column reports the gap between the lower and upper bounds. One observes that this gap is very small, especially for the first group of instances.

Finally, Tables 6, 7, 8 and 9 report the results obtained by using the branch-and-cut algorithm of CPLEX with and without the generation of user cuts, i.e., the cuts described in Section 4. In both cases, a maximum of 240 minutes of CPU time was allowed for the solution of each instance. In these tables, we report the best upper bound obtained, the CPU time in minutes, and the number of nodes explored in branch-and-cut tree. For the algorithm with user cut generation we also indicate the number of cuts added to the model. An asterisk preceding the CPU time indicates that the algorithm has reached the time limit before proving optimality.

On the one hand, one can observe that most instances from the first group (a) can be solved to optimality, and each of these instances contains 35 jobs. On the other hand, only half of the jobs from the third group (c) could be solved optimally although these contain only 20 jobs. This is an expected result since for the latter instances there is a larger initial gap between the lower and upper bounds. From Tables 6, 7, 8 and 9, one can see that the branch-and-cut algorithm with user cuts usually explores fewer nodes in the enumeration

Instance	n	Without reductions			With reductions			Upper	GAP
		Cons	Vars	LP	Cons	Vars	LP	bound	%
a-1	35	2665	6118	6145.18	2735	5536	6145.18	6171	0.420167
a-2	35	2665	6164	5492.91	2700	5596	5492.91	5538	0.820876
a-3	35	2665	6142	5484.00	2700	5555	5484.00	5495	0.200584
a-4	35	2665	6144	6054.18	2712	5577	6054.18	6070	0.261307
a-5	35	2665	6158	4842.55	2700	5546	4842.55	4900	1.186358
a-6	35	2665	6158	4753.35	2705	5579	4753.35	4810	1.191791
a-7	35	2665	6146	5062.95	2700	5563	5062.95	5077	0.277506
a-8	35	2665	6156	5087.26	2700	5561	5087.26	5092	0.093174
a-9	35	2665	6146	5584.76	2700	5550	5584.76	5648	1.132367
a-10	35	2665	6168	6213.16	2700	5581	6213.16	6269	0.898738

Table 2: Characteristics of the first group of instances

Instance	n	Without reductions			With reductions			Upper	GAP
		Cons	Vars	LP	Cons	Vars	LP	bound	%
b-1	25	1405	3158	9996.11	1430	2850	9996.11	10063	0.669160
b-2	25	1405	3158	9824.21	1430	2855	9824.21	9899	0.761283
b-3	25	1405	3168	9269.89	1430	2876	9269.89	9338	0.734744
b-4	25	1405	3164	12644.91	1430	2885	12644.91	12734	0.704552
b-5	25	1405	3168	11010.13	1444	2863	11010.13	11095	0.770836
b-6	25	1405	3160	10512.74	1447	2854	10512.74	10596	0.791991
b-7	25	1405	3162	11109.42	1442	2882	11109.42	11169	0.536302
b-8	25	1405	3172	11841.41	1433	2871	11841.41	11916	0.629908
b-9	25	1405	3164	10014.42	1430	2871	10014.42	10074	0.594942
b-10	25	1405	3164	9551.83	1435	2857	9551.83	9657	1.101046

Table 3: Characteristics of the second group of instances

tree even though a small number of cuts are generated. Furthermore, this algorithm could solve five more instances to optimality compared with the branch-and-cut of CPLEX.

In additional experiments, we tried to use the D_k^+ and D_k^- inequalities as well as the other inequalities presented in Section 4. The separation heuristic used for D_k^+ and D_k^- inequalities is that explained by (Fischetti and Toth (1997)) using $k \leq n-1$, $k \leq (n-1)/2$ and $k \leq (n-1)/4$. For most instances, these inequalities did not help and it seems that they in fact increased the number of fractional variables in the LP relaxation without significantly increasing the value of this relaxation.

7 Conclusion

This paper has introduced a new scheduling problem on a single machine in which the setup time is sequence-dependent and the processing times is time-dependent. Several formulations

Instance	n	Without reductions Cons	Vars	LP	With reductions Cons	Vars	LP	Upper bound	GAP %
c-1	20	925	2005	16334.17	965	1808	16334.17	16557	1.364195
c-2	20	925	2033	18842.22	958	1840	18842.22	19068	1.198266
c-3	20	925	2041	17052.65	960	1827	17052.65	17358	1.790631
c-4	20	925	2035	11700.32	951	1847	11700.32	11896	1.672433
c-5	20	925	2039	17169.00	954	1843	17169.00	17330	0.937737
c-6	20	925	2017	14315.30	965	1818	14315.30	14555	1.674432
c-7	20	925	2019	17390.85	965	1821	17390.85	17595	1.173893
c-8	20	925	2041	17824.18	953	1841	17824.18	18088	1.480124
c-9	20	925	2037	17820.76	952	1827	17820.76	18069	1.392982
c-10	20	925	2037	13871.88	951	1848	13871.88	14177	2.199558

Table 4: Characteristics of the third group of instances

Instance	n	Without reductions Cons	Vars	LP	With reductions Cons	Vars	LP	Upper bound	GAP %
d-1	20	925	2022	19596.63	964	1816	19596.63	20082	2.476803
d-2	20	925	2001	18814.38	965	1817	18814.38	19027	1.130093
d-3	20	925	2041	15470.00	949	1835	15470.00	16102	4.085326
d-4	20	925	2017	20227.87	965	1823	20227.87	20538	1.533182
d-5	20	925	2041	18809.95	960	1834	18809.95	19275	2.472362
d-6	20	925	2039	15371.61	953	1840	15371.61	15804	2.812913
d-7	20	925	2019	18123.00	965	1840	18123.00	18197	0.408321
d-8	20	925	2035	20090.12	957	1837	20090.12	20383	1.457831
d-9	20	925	2015	18338.63	965	1829	18338.63	18605	1.452508
d-10	20	925	2039	19471.00	953	1862	19471.00	19865	2.023522

Table 5: Characteristics of the fourth group of instances

of the problem have been proposed and we have shown that the LP relaxation lower bound associated with the three-index formulation dominates the lower bounds associated with the other formulations. We have also introduced several families of valid inequalities for this problem. Computational experiments have shown that the inequalities on intervals are particularly useful in reducing the number of nodes in the branch-and-cut tree as well as the CPU time.

Acknowledgments

This research was partially funded by the Natural Sciences and Engineering Research Council of Canada under grant 227837-04. This support is gratefully acknowledged.

	CPLEX B&C			B&C with user cuts			
Instance	Bound	CPU	Nodes	Bound	CPU	Nodes	Cuts
a-1	6171	164.702	116302	6171	101.753	52325	90
a-2	5538	*240.000	72257	5511	114.050	35903	88
a-3	5495	10.212	7727	5495	11.944	7383	55
a-4	6070	*240.000	176560	6070	189.049	103661	82
a-5	4900	*240.000	60283	4898	*240.000	54452	156
a-6	4810	*240.000	65542	4810	*240.000	59613	101
a-7	5073	4.772	3568	5073	10.855	6723	62
a-8	5092	0.100	37	5092	0.269	170	13
a-9	5597	99.180	23331	5597	15.621	3521	85
a-10	6269	*240.000	57319	6269	*240.000	50154	188

Table 6: Comparisons with CPLEX on the first group of instances

	CPLEX B&C			B&C with user cuts			
Instance	Bound	CPU	Nodes	Bound	CPU	Nodes	Cuts
b-1	10041	36.352	51895	10041	47.269	59154	87
b-2	9850	43.862	55455	9850	29.721	31351	97
b-3	9321	84.047	132834	9321	82.303	107100	80
b-4	12696	218.736	259124	12696	135.506	128666	144
b-5	11093	*240.000	252179	11095	*240.000	213597	142
b-6	10596	*240.000	255883	10596	*240.000	216331	136
b-7	11169	79.043	127494	11169	61.639	77001	80
b-8	11893	64.149	95723	11893	34.547	42752	101
b-9	10061	7.110	12607	10061	10.312	15301	55
b-10	9622	26.310	35361	9622	9.027	11916	71

Table 7: Comparisons with CPLEX on the second group of instances

References

B. Alidaee and N.K. Womer. Scheduling with time dependent processing times: Review and extensions. *Journal of the Operational Research Society*, 50:711–720, 1999.

N. Ascheuer, M. Fischetti, and M. Grötschel. Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. *Mathematical Programming*, 90:475–506, 2001.

T.C.E. Cheng and Q. Ding. Single machine scheduling with step-deteriorating processing times. *European Journal of Operational Research*, 134:623–630, 2001.

T.C.E. Cheng, Q. Ding, and B.M.T. Lin. A concise survey of scheduling with time dependent processing times. *European Journal of Operational Research*, 152:1–13, 2004.

	CPLEX B&C			B&C with user cuts			
Instance	Bound	CPU	Nodes	Bound	CPU	Nodes	Cuts
c-1	16557	46.239	64938	16557	41.315	44597	54
c-2	19047	10.467	18768	19047	16.731	21198	68
c-3	17358	*240.000	273441	17358	*240.000	209066	172
c-4	11896	*240.000	301017	11896	140.467	150268	56
c-5	17330	163.329	208134	17330	96.786	116198	90
c-6	14555	*240.000	324825	14555	*240.000	268541	97
c-7	17595	*240.000	323651	17595	213.302	217815	120
c-8	18088	*240.000	303959	18088	*240.000	263527	107
c-9	18069	*240.000	205573	18069	*240.000	133041	117
c-10	14177	*240.000	292877	14177	*240.000	264551	58

Table 8: Comparisons with CPLEX on the third group of instances

	CPLEX B&C			B&C with user cuts			
Instance	Bound	CPU	Nodes	Bound	CPU	Nodes	Cuts
d-1	20079	*240.000	315628	20079	182.041	196079	129
d-2	19027	4.557	9957	19027	6.811	10789	37
d-3	16102	*240.000	183688	16102	*240.000	191371	240
d-4	20514	54.555	88532	20514	55.302	65488	82
d-5	19275	*240.000	229588	19275	*240.000	169031	149
d-6	15804	*240.000	227717	15804	*240.000	190370	177
d-7	18197	0.197	501	18197	0.166	303	20
d-8	20354	21.590	35432	20354	24.226	31382	93
d-9	18605	13.480	23024	18605	15.713	20114	70
d-10	19809	214.030	215466	19809	200.550	185032	194

Table 9: Comparisons with CPLEX on the fourth group of instances

A.V. Donati, R. Montemanni, N. Casagrande, A.E. Rizzoli, and L.M. Gambardella. Time dependent vehicle routing problem with a multi ant colony system. Technical Report IDSIA-17-03, Istituto Dalle Molle di Studi sull'Intelligenza Artficiale, September 2003, 2003.

M. Fischetti and P. Toth. A polyhedral approach to the asymmetric traveling salesman problem. *Management Science*, 43:1520–1536, 1997.

M. Grötschel and M.W. Padberg. Polyhedral theory. In E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, editors, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, pages 251–305. Wiley, Chichester, 1985.

J.N.D. Gupta and S.K. Gupta. Single facility scheduling with nonlinear processing times. *Computers and Industrial Engineering*, 14:387–393, 1988.

A. Haghani and S. Jung. A dynamic vehicle routing problem with time-dependent travel times. *Computers and Operations Research*, 32:2959–2986, 2005.

A.A.K. Jeng and B.M.T Lin. Makespan minimization in single-machine scheduling with step-deterioration of processing times. *Journal of the Operational Research Society*, 55: 247–256, 2004.

A.S. Kunnathur and S.K. Gupta. Minimizing the makespan with late start penalties added to processing times in a single facility scheduling problem. *European Journal of Operational Research*, 47:56–64, 1990.

C. Malandraki and M.S. Daskin. Time dependent vehicle routing problems: Formulation, properties and heuristic algorithms. *Transportation Science*, 26:185–200, 1992.

C. Malandraki and R.B. Dial. A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. *European Journal of Operational Research*, 90:45–55, 1996.

C.E. Miller, A.W. Tucker, and R.A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the Association for Computing Machinery*, 7:326–329, 1960.

G. Mosheiov. Scheduling jobs under simple linear deterioration. *Computers and Operations Research*, 21:653–659, 1994.

G. Mosheiov. Scheduling jobs with step-deterioration: Minimizing makespan on a single and multi-machine. *Computers and Industrial Engineering*, 28:869–879, 1995.

G. Mosheiov. On flow shop scheduling with deteriorating jobs. Working Paper, School of Business Administration, Hebrew University, Jerusalem, 1996.

G. Stecco. *Exact and approximate algorithms for a production scheduling problem.* PhD thesis, University Ca Foscari of Venice, Italy, 2006. In preparation.

P.S. Sundararaghavan and A.S. Kunnathur. Sigle machine scheduling with start time dependent processing times: Some solvable cases. *European Journal of Operational Research*, 78:394–403, 1994.